Title: A framework for a composite application and a method of implementing a frame work for a composite application

Technical Field

The invention relates to a framework for a composite application and a method of implementing a frame work for a composite application.

5   Background to the Invention and Prior Art

Integration platforms, such as the SAP Enterprise Portal for integrating heterogeneous information technology (IT) assets, are known. Such platforms may include a framework for a composite application, and use *inter alia* object oriented programming (OOP) wherein objects, in particular business objects, are manipulated

10   in business applications.

An integration platform such as the SAP Enterprise Portal, it is known to provide generic services for collaboration with and/or between users, such generic services include subscriptions, rating, voting, discussion, comments and the like. Such generic

15   services have evolved out of document based content management systems. One problem with such services is that while they may be applicable to a particular business object, they are not necessarily directly applicable for other business objects. In particular, the generic services require customizing in order to be applicable to other business objects.

20

For example, depending on the business context, a system based conversation thread may be perceived by a user as a variety of processes: for example, a decision making process, a brainstorming session, an evaluation, a negotiation, a coordination, an audit trail, ...etc. Thus, a further problem with conventional platforms is that generic

25   services are unable to adapt to the actual work intent of the collaboration in terms of naming and function set.

It is an object of the present invention to address those problems identified above with respect to the prior art. In particular, it is an object of the present invention to provide a more explicit semantic to adapt generic services to a particular business application.

5   Summary of the Invention
According to a first aspect of the invention, there is provided a framework for a composite application, the framework including:
an object access layer operable to exchange data with a plurality of enterprise base systems and to present the data to a composite application through a uniform

10   interface;
a business modelling layer including a business object modeller operable to provide a user interface for constructing a business object; the framework further including:
a service layer operable to provide services to the composite application, wherein the service layer includes a collaboration services module operable to provide a plurality

15   of collaboration services to the composite application, wherein the object modelling layer is operable to directly link at least one of the plurality of collaboration services associated with the business object to the business object.
By directly linking associated collaboration services with business objects on the object modelling layer a more specific semantic is created, so that generic services are

20   adapted to the actual work intent of the collaboration. In particular, in this way, services can be specialised in accordance with a particular business object. Thus, providing an encapsulating functionality. In addition, further services are made available to a user. In particular, specific "flavours" of a generic service are made available depending on the particular business object.

25

According to a second aspect of the present invention, there is provided a method of implementing a composite application in a framework, the method comprising:
accessing an object to exchange data with a plurality of enterprise base systems and to present the data to a composite application through a uniform interface;

modelling a business object to provide a user interface for constructing a business object;

providing services to the composite application, wherein the providing services step includes providing a plurality of collaboration services to the composite application,

5   wherein the modelling step includes directly linking at least one of the plurality of collaboration services associated with the business object to the business object.

According to a third aspect of the present invention, there is provided a user terminal comprising means operable to perform any of the method of any of claims 9-16.

10

According to a fourth aspect of the present invention, there is provided a computer readable storage medium storing a program which when run on a computer controls the computer to perform the method of any of claims 9-16.

15   Specific embodiments of the invention are set forth in the dependent claims.

Brief description of the drawings
In order that the invention may be more fully understood embodiments thereof will now be described by way of example only, with reference to the figures in which:

20   Figures 1A and 1B are block diagrams illustrating an example integrated platform;
Figure 2 is a block diagram illustrating an example framework for a composite application;
Figure 3 is block diagram illustrating another example framework for a composite application;

25   Figure 4 is a block diagram illustrating design-time components for a composite application framework; and
Figure 5 is a block diagram illustrating components for guided procedure for a composite application framework.

Description of the preferred embodiments

The systems and techniques described here relate to a framework for developing and implementing applications in an enterprise management system. For example, a

5   framework may be used to develop and implement a composite application, which overlays an enterprise IT platform and uses it to implement processes that are not the core enterprise transactional processes. That is, a composite application may orchestrate a business process in synchronization with existing processes (e.g., native processes of enterprise base systems) and leverage existing investments in the IT

10   platform. Furthermore, composite applications may be run on a heterogeneous IT platform. In doing so, composite applications may be cross-functional. That is, they may drive business processes across different applications, technologies, and organizations. Accordingly, composite applications may drive end-to-end business processes across heterogeneous systems. Additionally, composite applications may be

15   combined with each other in order to enlarge the process coverage. Composite applications may also support semi-structured processes, tackle event-driven and knowledge-based scenarios, and support a high degree of collaboration in teams. In teams, for example, people may work on specific tasks in specific roles in specific teams. Composite applications may relate knowledge, structured information, and/or

20   unstructured information within the context of a business process and may be triggered by events, aggregate and contextualize information, and drive collaboration and transactions. Different applications supported by different frameworks may have any combination of these characteristics. Thus, different implementations of the framework may be used for developing and implementing various types of

25   applications.

FIGs. 1A-B illustrate an example integrated platform, also referred to as integrated enterprise management system 100. In system 100, clients 110 access data over a communication network 120 through a portal 130. Network 120 may be any appropriate type of communication network, such as, for example, a local area

30   network (LAN), a wide area network (WAN), an enterprise network, a virtual private

network (VPN), the Internet, and/or the Public Switched Telephone Network (PSTN). Clients 110 may be any machines or processes capable of communicating over network 120. In particular implementations, clients 110 may be Web Browsers and, optionally, may be communicatively coupled with network 120 through a proxy

5   server.

Portal 130 provides a common interface to program management services. In operation, portal 130 receives requests from clients 110 and generates information views 131 (e.g., Web pages) in response. The portal may implement a user-role based system to personalize the common interface and the information views 131 for a user

10   of one of clients 110. A user may have one or more associated roles that allow personalized tailoring of a presented interface through the generated information views 131.

In particular implementations, the portal may include *inter alia* one or more service interfaces to an enterprise management consolidation system 140. The service

15   interfaces may include an Internet Transaction Server (ITS) component, various connectors, such as a Java Connector, and a Business Intelligence platform.

Portal 130 communicates with enterprise management consolidation system 140, which consolidates multiple application services. Portal 130 receives information 141 from enterprise management consolidation system 140 for use in fulfilling the

20   requests from clients 110. Enterprise management consolidation system 140 provides integrated application services to manage business objects and processes in a business enterprise. The business objects and processes may be resources (e.g., human resources), development projects, business programs, inventories, clients, accounts, business products, and/or business services.

25   Enterprise management consolidation system 140 communicates with enterprise base systems 150 to obtain multiple types of information 151. Enterprise base systems 150 may include various existing application services, such as customer relationship management (CRM) systems, human resources management (HRM) systems, financial management (FM) systems, project management (PM) systems, knowledge

30   management (KM) systems (e.g., documents attached to a business object), business

warehouse (BW) systems, time management (TM) systems, and/or electronic file and
mail systems. The enterprise base systems may also include an integration tool, such
as an exchange infrastructure (XI), which provides another level of integration among
base systems.

5   Enterprise management consolidation system 140 may consolidate and integrate the
data and functionality of enterprise base systems 150 into a single enterprise
management tool. This enterprise management tool may include systems and
techniques to facilitate creation and execution of new applications within the
enterprise management consolidation system. These new applications may be

10   composite applications and may readily draw on the resources of enterprise base
systems 150 to cross over traditional enterprise application boundaries and to handle
new business scenarios in a flexible and dynamic manner, allowing rapid and
continuous innovation in business process management. A virtual business cycle may
be created using such cross-functional applications, where executive-level business

15   strategy may feed management-level operational planning, which may feed employee-
level execution, which may feed management-level evaluation, which may feed
executive-level enterprise strategy. The information generated at each of these stages
in the enterprise management cycle may be readily consolidated and presented by the
enterprise management consolidation system 140 using customized composite

20   applications. The stages may provide and consume determined services that may be
integrated across multiple disparate platforms.

Portal 130, enterprise management consolidation system 140, and enterprise base
systems 150 may reside in one or more programmable machines, which may
communicate over a network or one or more communication busses. For example,

25   base systems 150 may reside in multiple servers connected to an enterprise network,
and portal 130 and enterprise management consolidation system 140 may reside in a
server connected to a public network. Thus, system 100 may include customized,
Web-based, composite applications, and a user of the system may access and manage
enterprise programs and resources using these customized, Web-based, composite

30   applications from anywhere that access to a public network is available.

FIG. 1B further illustrates enterprise management consolidation system 140 for the example. System 140 includes a persistence layer 142 and one or more base system connectors 145. Base system connectors 145 enable data exchange and integration with enterprise base systems. Base system connectors 145, for example, may include

5 an Enterprise Connector (EC) interface, an Internet Communication Manager/Internet Communication Framework (ICM/ICF) interface, an Encapsulated PostScript® (EPS) interface, and/or other interfaces that provide Remote Function Call (RFC) capability.

Persistence layer 142 provides enterprise management consolidation system 140 with

10 its own database 143 and data object model 144. Database 143 and object model 144 provide a consolidated knowledge base to support multiple enterprise management functions, such as portfolio management, project execution, risk assessment, budgeting, scheduling, workforce planning, skills management, business forecasting, and capacity modeling, which could all be could be created as composite applications

15 149. Data object model 144 may represent a subset of data objects managed by the base systems. That is, not all of the data aspects tracked in the base systems need to be recorded in data object model 144. Data object model 144 may have defined relationships with data objects stored in the base systems; for example, certain objects in data object model 144 may have read-only or read-write relationships with

20 corresponding data objects in the base systems. These types of defined relationships may be enforced through the communication system built between persistence layer 142 and the base systems. Thus, persistence layer 142 may be used to effectively decouple application development and execution from the underlying base systems. Applications 149 take advantage of this decoupling from back-end systems to flexibly

25 integrate existing systems and new functional components into business processes. Furthermore, the applications may drive business processes across different platforms, technologies, and organizations.

Applications 149 may be created using a set of tools that enable efficient application development. The tools may enable efficient application development by providing

8

application patterns that support model-driven composition of applications in a
service-oriented architecture.

An object modeling tool 146 enables creation of new business objects in the
persistence layer 142 by providing a mechanism to extend data object model 144

5   dynamically according to the needs of an enterprise. The object modelling tool 146
includes a module 132 operable to derive at least one object specific service from at
least one generic collaboration service. The module 132 is further operable to derive
an object class specific service from at least one generic collaboration service. A
process modeling tool 147 enables creation of new business workflow and ad hoc

10   collaborative workflow. A user interface (UI) tool 148 provides UI patterns that may
be used to link new objects and workflow together and generate standardized views
into results generated by applications 149. Object modeling tool 146, process
modeling tool 147, and UI tool 148, thus, may be used to build the components of
applications 149 to implement new enterprise management functions without

15   requiring detailed coding activity.

Process modeling tool 147 may include guided procedure templates with pre-
configured work procedures that reflect best practices of achieving a work objective
that is part of a larger cross-functional application scenario. Such a work procedure
may include contributions from several people, creation of multiple deliverables, and

20   milestones/phases. Moreover, whenever an instantiated business object or work
procedure has lifetime and status, the progress and status of the object or work
procedure may be made trackable by the process owner or by involved contributors
using a dashboard that displays highly aggregated data. A dashboard and a page that
provides access to status information about ongoing work, such as a personalized

25   work place, may be two UI patterns that are provided by UI tool 148.

Moreover, there may be other UI personalizations. For example, if there is a concept
of personalized items, such as, for example, objects, recent objects, related objects, or
preferred objects, then an object picker UI pattern, provided by UI tool 148, may be
included to let users pick their favorite object directly. Such objects can be used to

30   determine the context using a context modeller, as shown in Figure 5.

Composite application scenarios may provide related information to the user when possible, and some parts within a larger application scenario may define what kind of related information is to be offered. Heuristics may be used to identify such relatedness, such as follows: (1) information that is related to the user due to explicit

5   collaborative relationships such as team/project membership or community membership; (2) information that is similar to a given business object in a semantic space based on text retrieval and extraction techniques; (3) recent objects/procedures of a user; (4) other people performing the same or similar activity (e.g., using same object or procedure template having the same workset); (5) instances of the same

10   object class; (6) next abstract or next detailed class; (7) explicit relationships on the organizational or project structure; (8) proximity on the time scale; (9) information about the underlying business context; and/or (10) information about the people involved in a collaborative process.

15   Composite applications also may include generic functionality in the form of ControlCenter Pages that represent generic personal resources for each user. These may refer to the following pages where appropriate: (1) MyOngoingWork page: provides access to status information about ongoing work of a user (Ongoing work may refer to the state of business objects as well as guided procedures); (2) MyDay

20   page: lists today's time-based events that are assigned or related to a user; (3) MyMessageCenter page: displays pushed messages and work triggers using a universal inbox paradigm with user selected categorical filters; and/or (4) MyInfo: provides access to personal information collections (e.g., documents, business objects, contacts) including those located in shared folders of teams and communities of which

25   the user is a member. The page may also provide targeted search in collaborative information spaces such as team rooms, department home pages, project resource pages, community sites, and/or personal guru pages.

FIG. 2 illustrates a framework 200 for a composite application. In general, framework 200 leverages and enhances underlying enterprise base systems 290,

30   which could include an XI, supporting business transaction systems such as CRM,

HCM, and PLM, Knowledge Management Warehouse (KW), and BW, with tools, content, and guidelines to provide a foundation for developing and executing composite applications.

As discussed previously, composite applications typically implement new or additional

5   processes, as opposed to the core transactional processes, in an existing IT landscape. Composite applications may also support semi-structured processes, tackle event-driven and knowledge-based business scenarios, and support collaboration in teams. In particular implementations, composite applications may support the Java stack.

10   As illustrated in FIG. 2, framework 200 includes an object access layer (OAL) 210, a service layer 220, a user interface (UI) layer 230, and a metadata repository 240. OAL 210 manages interaction between composite applications and enterprise base systems 290. In doing so, OAL 210 provides a uniform interface for composite applications. Thus, OAL 210 reduces the knowledge needed for a composite

15   application developer about the source of data because OAL 210 sits on top of and embraces different connectivity technologies. Coding and configuration data for OAL 210 may be automatically generated, at least in part, by business object metadata in repository 240. Furthermore, OAL 210 allows for local persistency (e.g., connectivity to a local database such as an application database 250 to store data). Data

20   synchronization and replication of remote data (e.g., data in back-end systems) into the local persistency database may be supported. For an application sitting on top of layer 210, the source of the data may be completely transparent, which may assist in keeping application logic stable since the application is, at least for the most part, not affected by underlying systems. In some implementations, OAL 210 includes

25   extensions to document management or content management that allow business objects to use the functionality for documents.

Service layer 220 provides services for business objects in layer 210. In general, services for business objects are common procedures that users need to interact effectively with the objects. Service layer 220, for example, may include generic

30   services including generic collaboration services, guided procedure services, and/or a

container for application services. By separating the services from the business objects, the services may be more readily reused across business objects. As will be described hereinbelow with reference to Figure 4 and shown in Figure 1B, in an embodiment of the present invention object specific collaboration services are derived

5   in the object modelling layer 146, 410,

UI layer 230 provides user interfaces that allow a user to interact with composite applications. In particular implementations, UI layer 230 provides pattern components, such as, for example, a dashboard, a search bar, a browse and collect function, an object editor, and phases for a guided procedure, as building blocks for

10   user interfaces. UI layer 230 may also decouple application logic from the UI. As shown, UI layer 230 accomplishes this by having a separation of the business objects, which are in the object access layer 210, and application services, which are in service layer 220, from the user interface elements, which are in UI layer 230. This allows UI components to be reused in different application contexts. This also allows business

15   objects and application services to be visualized differently according to the specific equipments of a certain use case. UI layer 230 may also leverage the metadata information on business objects and services through metadata-driven UI-generation and configuration. The metadata approach allows for ready adaptability to alternative screens depending on the end users needs (e.g., in different industries).

20   UI layer 230 may additionally allow integration (e.g., binding) into OAL 210 to access business objects, business services, and metadata. Thus, UI components may be connected to business objects in OAL 210. UI layer 230 may support any appropriate type of user interfaces, such as, for example, a user interface composed of pattern-based components and/or freestyle components with interfaces to the user interface

25   components -- this user interface will discussed in more detail below -- or Java Server Pages (JSPs) from Sun Java Server Pages (JSPs) from Sun.

Metadata repository 240 stores the content of the composite application (e.g., specific business objects, information about services, and, eventually, processes) and makes the metadata information available at run-time, if needed. The repository may allow

12

different metamodels to be created (the model for business objects being one of them) and to persist the metadata.

As mentioned previously, attached to framework 200 is application database 250. Database 250 provides a central repository for available business objects. An example

5   of data in repository 250 includes database tables for a business object. The data may be added to, changed, and/or deleted. Data may also be stored in KW, BW, or an XI system. As discussed, framework 200 provides a set of standard services that enables application developers to make use of the data. In particular, according to an embodiment of the invention, specific services are directly linked to a particular

10   business object in accordance, for example, with the context as modelled by context modeller 724, shown in Figure 5. Based on the central repository for objects, metadata data about objects is stored in metadata repository 240. This metadata enables generic services like automatic generation of default UIs, object access interface, data access methods, persistency, and mappings.

15   Framework 200 may be implemented using readily available technology. For example, the framework may be implemented using mySAP technology components. In particular implementations, the components may include an SAP Web Application Server (WAS) to run the applications, an SAP Enterprise Portal to render the applications, an SAP KW to handle unstructured information sources, pattern-based

20   components and/or freestyle components with interfaces to the UI components to design UIs and to provide J2EE and ABAP run-time integration, an SAP BW to provide reporting and analytics, data mining, and planning and simulation, SAP Business Process Management (BPM), an SAP Exchange Infrastructure (XI) to provide shared integration knowledge separate from applications, and SAP Web

25   services to offer business functionality over the Internet.

In one general aspect, framework 200 allows composite applications in which object specific collaboration services are provided to work with existing system landscapes. The framework accomplishes this by decoupling composite applications from the underlying enterprise platform, which includes enterprise base systems and deriving

30   and linking at the object modelling layer 146, 410 specific services to business objects.

This decoupling may involve providing communication to back-end systems via a central interface and providing a back-end-independent object model. The latter may be implemented so that the data from the source systems may be transformed into a unified structure.

5   Examples of the types of business processes supported by the framework include, but are not limited to product innovation including submitting a new idea, concept development, employee productivity, enterprise change management and enterprise service automation. Enterprise change management may support enterprises when merging, splitting, acquiring, spinning off, or reorganizing. Product innovation may

10   support the life cycle of a product, including the prenatal phase of collecting ideas and consolidating them into concepts, the market launch phase, and the end of life. In doing so, the resources of a PLM and CRM may be drawn upon. Employee productivity aims to increase employee productivity, decrease costs, and increase employee satisfaction. Key functions may include manager self services, employee

15   self services, expert finders, e-procurement, and e-learning. ERM and B2E resources may be drawn upon to accomplish these tasks. Enterprise service automation provides administration and monitoring functions as well as evaluation tools to facilitate project success. An example of this is the setting up of projects and the staffing with people with the required skills and availability. Additional application

20   families may also be created.

FIG. 3 is a block diagram of a composite application framework 300 illustrating details of one potential implementation. As illustrated, framework 300 includes design-time components 310, run-time components 320, and a metadata repository 360, which is shared by the design-time components and the run-time components. In

25   general, design-time components 310 are responsible for developing composite applications that are executed by run-time components 320. Design time components 310 include, for example, the object modelling layer 146, 410 including the object modeller.

In more detail, design-time components 310 provide a repository and user interface

30   for modeling and generating business objects, business services, business processes,

user interfaces, and/or any other appropriate portions of a composite application. A business object, for example, may be an employee, a product, a plant, or any other semantic representation of a real-world entity. A business service is an action taken on a business object. Changing the price or category of a product are examples of

5   services for a business object that represents a product. As another example, gathering input from employees and customers, who may themselves be represented by business objects, for a new product idea are examples of business services. Putting services together in a proper combination produces a business process. A composite application is typically composed of business objects, business services, and/or

10   business processes.

As illustrated, design-time components 310 include application modeling tools 312, application generators 314, and, in part, metadata repository 360. Modeling tools 312 may be used for modeling business objects, business services, business processes, user interfaces, and the like. A separate modeling tool may be used for each of the

15   composite application portions. Furthermore, modeling tools 312 may be used for integrating business objects, business services, business processes, user interfaces, and the like. Thus, framework 300 may support model-driven composition of composite applications, allowing for development with little or no programming effort. The metadata about business objects, business services, business processes, and/or

20   other application portions is stored in metadata repository 360. Thus, an application portion may be modeled as well as the origin of the data, whether in a local database, remote database, or mixture. Generators 314 are used for generating actual code from the portions modeled by modeling tools 312. To accomplish this, the generators may use templates that are stored in metadata repository 360. Driven by the

25   metadata in repository 360, the generators may automatically create Java classes (e.g., for use in run-time components 320) and also configuration files (e.g., to adjust UI patterns to a certain business object). Thus, the connectivity to back-end systems and the application persistency may be generated, as well as a default user interface. The generators may also generate interfaces for application services, data access logic,

30   and persistency.

Run-time components 320 provide the run-time environment for business objects, business services, business processes, user interfaces, and the like, as well as data access abstraction. As illustrated, run-time components 320 include an object access layer 330, a service layer 340, a UI layer 350, and, in part, metadata repository 360.

5   Run-time components 320 also use an application database 370, which stores data tables for executing applications.

Object access layer 330 manages interaction between composite applications and enterprise base systems 390. In doing so, layer 330 reduces the knowledge needed for the application developer about the source of data because layer 330 sits on top of and

10   embraces different connectivity technologies. Thus, layer 330 provides a uniform interface for composite applications. As such, layer 330 may act as a dispatcher to provide access to a variety of data sources. As illustrated, layer 330 leverages a message-based platform 390a that includes an XI with connectivity to underlying applications like CRM, HCM, and PLM, a knowledge management warehouse (KW)

15   390b, and a business intelligence warehouse (BW) 390c, and manages the persistency in application database 370. For data access abstraction, the fact that layer 330 sits on top of and embraces different connectivity technologies allows routing to a variety of different data sources. Furthermore, layer 330 allows for local persistency (e.g., connectivity to a local database such as application database 370 to store data).

20   Additionally, data synchronization and replication of remote data (e.g., data in back-end systems) into the local persistency database may be supported. The data may be transferred and transformed into the local persistency. For an application sitting on top of layer 330, the source of the data may be completely transparent. For example, some applications underlying layer 330 might provide services (e.g., Web services),

25   allowing the data to be read remotely, and some applications may not provide such access.

In certain implementations, layer 330 includes extensions to document management or content management that allow business objects to use the functionality for documents. For example, taxonomies for business objects, transparent indexing of

30   TREX for structured and unstructured objects, and subscription services for

dependent objects independent of the repository where the objects reside may be provided. Layer 330 may also provide transaction support, in as far as the transaction concept is also supported by concerned source systems, a metadata interface, allowing an application to be dynamically configured at run-time, and

5   subscription services (e.g., J2EE publish and subscribe).

As a further example, layer 330 may facilitate application building by configuration. This may be accomplished by providing standard interfaces with well-defined semantics, which allows components to be combined in a meaningful way since the semantics of the components' interfaces is known, and allowing objects to participate

10 in a collaborative context, (e.g., chat room) just by implementing certain interfaces. Service layer 340 provides services for business objects in layer 330. In general, services for business objects are common procedures that users need to interact effectively with the objects. A service layer may also provide other types of services, such as, for example, UI-related service and/or collaboration services. Furthermore

15 the service layer may provide integration of external services.

As illustrated, service layer 340 includes generic services 342, collaboration services 344, guided procedure services 346, and a container for application services 348. By separating the services from the business objects, the services may be more readily reused across business objects.

20 As its name implies, generic services 342 provides a set of standard services for parts of an application. Moreover, the services may be used across applications. Generic services 342 may also provide namespace and packaging concepts. The services are typically not bound to a portion of an application, but are available to all portions. Examples of generic services include print services, value help services, authorization,

25 personalization, and voice enablement. An example of a value help service is the filling of drop down boxes in user interfaces; the service is able to determine what the possible entries are for boxes and to populate the boxes therewith.

Collaboration services 344 provides the ability to link semantic objects to business objects. Semantic objects typically provide a set of generic services, like classification,

30 notification, subscription, feedback, search, retrieval, rating, time-based publishing,

state-based publishing, and security model. In addition, relations between semantic objects may be supported. For example, a team could be assigned to a task, and people could be assigned to the team. Moreover, a room could be created for that task, to keep people and documents together. Semantic objects such as document, folder,

5   room, task, meeting, user, and discussion may be accessible via layer 330. Semantic objects may also be available in a variety of other ways. For example, semantic objects may be included in layer 330 as business objects, and/or individual services of semantic objects may be included in layer 340.

Collaboration services 344 extends the semantic object concept by making the

10   functionality of semantic objects available for business objects (e.g., notification, subscription, etc.). Thus, services 344 provides collaboration context for a business object. Services 344 may automatically manage the relations between business objects and semantic objects. In addition, new kinds of relations may be supported: for example, relations between business objects and semantic objects. Thus, a task or

15   a team may be assigned to a specific product, people may be assigned to the task, and so on. Furthermore, special collaborative services may be provided for semantic objects, such as scheduling and assignment functions for tasks and inviting, splitting, and closing functions for discussions. In particular implementations, a suite of collaboration services may be provided without the need to deal with KM specific.

20   These services may also be made available for composition applications.

Furthermore, the relation between the business objects and the semantic objects may be maintained.

The collaboration provided by collaboration services 344 may be semi-structured processes. A common understanding of a business process may be reflected by a

25   predefined collaboration scenario. On the other hand, the business process may be adaptable to different enterprise's needs. To support this, differing scenarios may be built with minimal programming.

According to an embodiment of the present invention, collaboration services are

30   directly linked in the object modelling layer to business objects. Further, the object

modelling layer includes a module 132 operable to derive object specific services from generic collaboration services. As described each business object is a specific instance of an object class. Module 132 is further operable to extend the object class by adding a generic collaboration service and operable to derive n object class specific service

5   from at lest one generic collaboration service. For example, with the composite application framework (CAF), business objects are modelled. According to an embodiment of the present invention, besides the modelling of the business object data structure and core lifecycle services such as "create", "edit", "delete", the object designer, for example, the user, can assign pre-configured or tailored collaboration

10   services to the business object that represent best practice collaboration scenarios around the particular object. Such pre-configured or tailored services may on the one hand be derivatives of generic collaboration services, such as "feedback", "rating", "voting", "discussion" and "comments", etc. A derivative of a generic collaboration service is a generic collaboration service wherein a more specific functionality of the

15   service has been derived. Such pre-configured or tailored may, alternatively, or in addition, be Guided Procedure templates that represent more complex work procedures. Further, such pre-configured or tailored collaboration services may include simple collaborative actions that represent one-step ad hoc people-to-people coordination patterns.

20

For example, according to an embodiment of the invention, on the business object: "Product Concept", the object designer can attach an "Innovation Talk" collaboration service derived from the generic service "discussion thread", or a "plus/minus rating" service derive from the generic collaboration service "voting". For the business object:

25   "customer", the object modelling layer, the object modeller is operable to associate the Guided Procedure template "Investigate financial solvency". In a further embodiment, for the business object: "Sales Process", the object modelling layer is arranged to link the ad hoc service "Request Request for Payment (RFP)".
       In a further embodiment, for the collaboration service "subscription", the object

30   modeller is operable to predefine events that are exposed to the end user for

subscription. In a sales process this may be "Notify me when new competitive situation arises" or "When scope of sales processes changes".

In a further embodiment, all or at least some of the collaboration services can be defined on a meta data level using logical services and data schemes to allow the

5   connection to any engine that provides such collaboration services. In one embodiment, this is achieved by a meta data modeller operable to map between the web service proxies and the generic collaboration services and further operable to look up target system addresses flexibly in the system landscape directory.

In this way, collaboration services and business applications are integrated. In

10   particular, by providing collaboration services on top of business applications.

The expression "collaboration service" may, in the art, also be referred to as "collaborative service", "collaboration process", or "collaborative process" and the like.

Guided procedure services 346 allows business objects to participate in guided

15   procedures. A guided procedure is a series of steps, often involving human interaction, that should be performed during the execution of a composite application. A guided procedure, which is a type of workflow, is typically one that is common to a variety of applications and, thus, may be reused. To provide guided procedures, services 346 may provide pre-defined building blocks for process workflow and pre-

20   defined actions.

Application services container 348 is used to implement model specific services for one or more business applications. Although generic objects, generic services, and/or processes may be generated for an application, some business logic is too specific to be implemented generically.

25   UI layer 350 includes a UI framework 352. Framework 352 provides pattern components as building blocks for user interfaces. Examples of pattern components include a dashboard, a search bar, a browse and collect function, an object editor, and phases for a guided procedure. These components may serve to increase efficiency of UI development because they are reusable and may serve to keep training costs down

30   by providing a standard look and feel for the composite applications. Furthermore,

the components readily provide a UI for composite application objects and services and allow a default UI to be automatically generated for displaying, creating, and changing business objects.

Framework 352 may also decouple application logic from the UI. As shown,

5   framework 352 accomplishes this by having a separation of the business objects and application services from the user interface elements. This allows UI components to be reused in different application contexts. This also allows business objects and application services to be visualized differently according to the specific equipments of a certain use case.

10   UI framework 352 may also leverage the metadata information on business objects and services through metadata-driven UI-generation and configuration. The metadata approach allows for ready adaptability to alternative screens depending on the end users needs (e.g., in different industries). UI framework 352 may additionally allow integration (e.g., binding) into layer 330 to access business objects, business

15   services, and metadata. Thus, UI components such as patterns and freestyle may be connected to business objects in layer 330. In accomplishing this, framework 352 may provide the necessary metadata at design-time and manage the access to the according service providers at run-time.

UI framework 352 may support any appropriate type of user interfaces. For example,

20   the UI framework may support a user interface composed of pattern-based components and/or freestyle components with interfaces to the user interface components -- this user interface will discussed in more detail below -- or Java Server Pages (JSPs) from Sun. UI framework 352 may also support a Java front-end and ABAP back-end, a Java front-end and Java back-end, or any other appropriate

25   combination of front-end and back-end. The framework may additionally provide a construction kit for complex components and applications and configuration of patterns via XML, URL, or other appropriate technique.

Metadata repository 360 stores the content of the composite application (e.g., specific business objects, information about services, and, eventually, processes) and makes

30   the metadata information available at run-time (if needed). The repository may allow

different metamodels to be created (the model for business objects being one of them) and to persist the metadata. For specific purposes, additional repositories, such as, for example, a portal content directory (PCD), which may contain portal specific pieces of an application (e.g., views, pages, roles), may be required.

5   As mentioned previously, attached to framework 320 is an application database 370. Database 370 provides a central repository for available business objects. An example of data in repository 370 includes database tables for a business object. The data may be added to, changed, and/or deleted. Data may also be stored in KW, BW, or an XI system.

10   Based on the central repository for objects, metadata data about objects is stored in metadata repository 360. This metadata enables generic services like automatic generation of default UIs, object access interface, data access methods, persistency, and mappings.

       In particular implementations, modelers 312 and generators 314 generate the

15   business objects used in layer 330. Modelers 312 and generators 314 also facilitate the creation of business object metadata and its storage in metadata repository 360. The modelers and generators may be relatively easy to use because they are restricted to a particular purpose. Additionally, they may help to ensure the consistency of the metadata according to the capabilities of layer 330.

20   In particular implementations, the composite application portions may be implemented as Enterprise Java Beans (EJBs). In other implementations, the design-time components may have the ability to generate the run-time implementation into different platforms, such as J2EE, ABAP, or .NET.

       Components 310 may also support a variety of specific features needed for business

25   objects, such as time-dependent attributes or organizational-unit-dependent attributes, like product attributes, which differ from plant to plant. The components may not only generate the classes and the coding, but may also create the database tables and the interfaces to the UI, including the relevant metadata. So, after modeling, there may be a generation step that provides a stack of services for one

22

business object, including the UI down to the database tables and proxies for remote access.

5   Framework 300 may be implemented using readily available technology, for example, in the same or similar way to framework 200 discussed above. In particular, an SAP WAS may include a J2EE engine, SAP IDE, Universal Workflow, and Deployment Service. The WAS may also include a pattern-based and freestyle-based user interface development and interface module. Also, an SAP Enterprise Portal may
10   provide unified access to applications, information, and services by using views, roles, pages, worksets, top-level navigation, and KM. This enterprise portal also provides login management and user management. For KM, unstructured information consists of collaboration and content management. For collaboration, KM enables team-driven business processes, synchronous and asynchronous applications,
15   groupware integration, calendars, bulletin boards, threaded discussions, and collaboration rooms. For content management, KM handles documents, feedback, rating, publishing, subscription, document workflow, versioning, archiving, indexing, searching, and taxonomies. SAP BPM may cover life cycles (e.g., design, development, deployment, and change). An SAP XI may provide external and internal integration
20   of system and connectors to various systems such as Oracle, Siebel, Peoplesoft, and SAP. The SAP XI may be based on Web services, JAVA, and XML standards. SAP Web services may provide a service provider, service handler, and service user. Additionally, an SAP BW may be used.
       Moreover, the KM and collaboration functionality may embedded in applications, not
25   only in separate pages in the portal. Furthermore, any general development environment may be used. For example, the development environment could include Java, with EJB 2.0, JDO, Java persistency, and Java application logic, Advanced Business Application Programming (ABAP), and Web services. Existing ABAP components may be integrated via Java connector calls. In particular

implementations, the complete Java stack could be used. Furthermore, Web service technology may be used for remote access.

In one general aspect, framework 300 allows composite applications to work with existing system landscapes. The framework accomplishes this by decoupling

5   composite applications from the underlying enterprise platform. This decoupling may involve providing communication to back-end systems via a central interface and providing a back-end-independent object model. The latter may be implemented so that the data from the source systems may be transformed into a unified structure. This may also allow successive installation, activation, and use of different

10   applications, which may reduce entry costs.

Examples of the types of business processes supported by the framework 300 include those supported by framework 200.

FIG. 4 illustrates design-time components 400 for a composite application framework. Design-time components 400 could be representative of design-time components 310

15   in FIG. 3. As illustrated, design time components 400 include a business object modeler 410, a business object generator 430, and a metadata repository 450. Note that metadata repository 450 is also a run-time component.

Business object modeler 410 includes an Integrated Development Environment (IDE) application program interface (API) 411, an object modeler 412, and a relation

20   modeler 413. IDE API 412 allows modeler 412 to be integrated into an Eclipse IDE, which supports the modeling of the business object by object modeler 412. For example, the integration supports generation of business objects as EJBs, interfaces for application services, default user interfaces, data access logic, and persistency. Relation modeler 413 allows the modeling of relations between modeled objects. For

25   example, a sales order could be composed of a customer, a product, and a price. Relation modeler 413, therefore, allows for the modeling of the relations between these items. In operation, for instance, if a user interface is generated for a sales order, the semantics for each field in the sales order may be identified. Additionally, a connection to the value help function may be facilitated.

Modeler 410 also includes a metadata API 414 and a generation API 415. Metadata API 414 allows object modeler 412 to store and access business object metadata in metadata repository 450 and relation modeler 416 to store and access business object relation metadata in metadata repository 450. Generation API 422 allows a business

5   object to communicate with generator 430 for code generation.

Generator 430 includes a generator framework 432, a persistency generator 434, an EJB 436 generator, a UI adapter generator 438, a Web service generator 440, and a metadata API 442. Generator framework 432 may also be integrated into the Eclipse IDE.

10   To generate a business object, generator 430 may use templates in metadata repository 450 and code them with object metadata and relation metadata in the repository. Generator 430 may also generate the data persistency for the business object, and generate the actual business object, an EJB in this instance. Generator 430 may additionally generate user interfaces for the business object and any

15   necessary Web services.

The templates may be generic. In particular implementations, the generators automatically create Java classes (e.g., for the implementation of the object access layer), JDO tables, EJBs, and configuration files, to adjust UI patterns to a certain business object, for example. Thus, the connectivity to back-end systems and the

20   composite application persistency is generated as well as a default User Interface. Furthermore, UI adapters for a UI development and interface module and, if necessary, Web services may be generated. The output of such a process may be real working code in the object access layer of the run-time components.

One example is the generation of a run-time implementation of a business object in an

25   object access layer. The generator reads the business object metadata from the repository and generates the JDO persistency, the connectivity to the XI, the KW and/or the BW (e.g., by using proxies), the generic methods, and the basic UI. For this coding, templates (e.g., for services) or XML-templates (e.g., for JDO persistency) are used where business object specific coding or XML is added, and the result is stored as

30   complete code or complete XML.

Metadata repository 450 includes object metadata 452, relation metadata 454, and code generation templates 456. As mentioned previously, the information in object metadata 452 and relation metadata 454 may be used to code templates 456 to generate a business object.

5

There are a variety of types of methods that may be created for business objects. One example is lifecycle methods (e.g., create, update, etc.). Another example is standard methods that do not require coding. An example of such methods are those that allow business objects to participate in collaboration (e.g., subscribe, notify, discuss, etc.).

10   These methods may be automatically called when a certain action is carried out (e.g., when a business object should be updated). In particular, these collaboration services may be specialised in accordance with a particular business object, as described hereinabove. Another type of method is one that is specific to the composite application. These methods may be directly called by the applications.

15   FIG. 5 illustrates components 700 for guided procedures for a composite application framework. As mentioned previously, a guided procedure is a series of steps, often involving human interaction, that should be performed during the execution of a composite application. A guided procedure is typically one that is common to a variety of applications. As illustrated, components 700 may be classified into design-

20   time components 710 and run-time components 750, except for a metadata repository 790, which is part of both. Design-time components 710 may be used to generate run-time components 750.

Design-time components 710 include a modeler 720 and a generator 730. Modeler 720 includes a process modeler 722, a pattern modeler 725, and an action modeler 726.

25   Process modeler 722 includes a workflow modeler 723 and a context modeler 724. As their names imply, workflow modeler 723 allows process workflow for a guided procedure to be modeled, and context modeler 724 provides context definition. That is, context modeler 724 allows relations between other processes to be defined. As an example of this, an application may have more than one way of being activated ,

30   Intranet Web-based form versus remote voice control, for example. Context modeler

724 is responsible for making sure that both activation mechanisms are associated
with the application. In accordance with the context modelled by the context modeller
724, a specific collaboration service can be derived.

Pattern modeler 725 provides workflow patterns (e.g., delegation and approval) for
5   workflow modeler 723, and action modeler 726 provides actions for workflows.
Modeler 720 also includes a metadata API 727, which provides access to the data in
metadata repository 790. Thus, access to meta data regarding guided procedures is
available.

Generator 730 includes a template generator 731, a state chart generator 733, a
10   pattern generator 735, an action generator 737, and a metadata API 739. Templates
describe a workflow that may be may be implemented using workflow patterns.
Workflow patterns contain actions that must be accomplished to complete the
workflow and, hence, part of the template. Thus, a pattern may be viewed as an
abstraction of an action, and a template may be viewed as an abstraction of work flow
15   pattern.

For example, a template could describe a workflow for ordering a product , a
computer, for example. The template may specify a workflow pattern for obtaining
manager approval. The pattern would have certain actions that need to be
undertaken. An example of an action could be finding the names of the employee's
20   managers. The approval pattern, moreover, could be used for different templates.

As their names imply, template generator 731 generates templates, state chart
generator 733 generates state charts, pattern generator 735 generates patterns, and
action generator 737 generates actions for the run-time environment. Metadata API
739 provides access to the metadata in metadata repository 790.

25   Metadata repository 790 includes templates 792, workflow patterns 794, actions 796,
and metadata 798. The templates, patterns, actions, and metadata may be accessed
by generator 730 to produce a guide procedure.

Run-time components 750 provide instantiation for guided procedures, producing
instances 752. Procedural navigation and integration may be provided in a Universal
30   Worklist (UWL).

Run-time components 750 also include object access services 760, context sharing service 762, content services 764, portal connector service 766, KM service 768, workflow service 770, and metadata services 772. Object access services 760 allows objects in an object access layer to be accessed. Context sharing service 762 provides

5   context to a workflow. For example, when a user accesses a workflow, context sharing service 762 provides a link to the proper portions of the workflow. For instance, many workflows involve inboxes, where new tasks for the workflow may be sent. The inbox may provide a link to the proper portion of the workflow if the context is known. Content services 764 provide services for executing functions based on generic calls.

10   For example, a workflow may need an application , a composite application, an HRM application, or a CRM application, for example , to be initiated. By making a generic call to content services 764, the application may be initiated. Content service 764 may support integration with an application and/or a user interface. Portal connector service 766 provides a connection service to a portal. KM service 768 provides a

15   connection service to a KM module. Workflow service 770 provides a connection service to an ad-hoc workflow. This workflow may be very user-centric, allowing the assignment of not only tasks handled by transactions in business systems, but also tasks that require user handling (e.g., compose e-mail). Metadata services 772 provides a connection to metadata repository 790.

20   Components 700 may have a variety of features. For example, the components may provide context mapping for building blocks, and a user profile may be automatically used and updated. In certain implementations, ad-hoc administrations of running workflows may be supported and guided procedures may be monitored and analyzed.

25   Figure 6 is a flowchart showing a method 60 of implementing a composite application in a framework in accordance with an embodiment of the invention. According to an embodiment of the present invention, there is provided a method of implementing a composite application in a framework 200, 300, the method comprising:
       accessing an object to exchange data with a plurality of enterprise base systems and

30   to present the data to a composite application through a uniform interface (step 62);

modelling a business object to provide a user interface for constructing a business object (step 64); providing services to the composite application; wherein the providing services step includes providing a plurality of collaboration services to the composite application (step 66), wherein the modelling step includes directly linking at least one

5   of the plurality of collaboration services associated with the business object to the business object (step 68). If the session is ending (step 70) the method ends (step 72). If the session is not ending, the method returns to step 62.

In particular, the providing a plurality of collaboration services step may include providing at least one generic collaboration service. Further, the modelling step may

10   include deriving at least one object specific service from the at least one generic collaboration service. Yet further, the method may include modelling a process, wherein the modelling a process step includes modelling a context, wherein the modelling step includes deriving an object specific service from the at least one generic collaboration service on the basis of the modelled context. Yet further, each

15   business object may be a specific instance of an object class, wherein the modelling step may include extending the object class by adding a generic collaboration service and deriving an object class specific service from the at least one generic collaboration service. In particular, the modelling step may include specializing the at least one generic collaboration service in accordance with the object class. Further, the

20   modelling step may be carried out by a design time component. The method may include the further step of providing a user interface layer operable to provide user interface patterns that facilitate information exchange between the composite application and a user.

25   The computational aspects described here can be implemented in digital electronic circuitry, or in computer hardware, firmware, software, or in combinations of them. Where appropriate, aspects of these systems and techniques can be implemented in a computer program product tangibly embodied in a machine-readable storage device for execution by a programmable processor, and method steps can be performed by a

programmable processor executing a program of instructions to perform functions by operating on input data and generating output.

5   The invention may also be implemented in an article of manufacture with a computer usable medium having computer readable instructions embodied therein for providing access to resources available on that computer, the computer readable instructions comprising instructions to cause the computer to perform the steps of a method according to the invention. The invention may also be implemented as a computer program for running on a computer system, at least including code portions for

10   performing steps of a method according to the invention when run on a computer system or enabling a general propose computer system to perform functions of a filter device according to the invention. Such a computer program may be provided on a data carrier, such as a CD-ROM or diskette, stored with data loadable in a memory of a computer system, the data representing the computer program. The data carrier

15   may further be a data connection, such as a telephone cable or a wireless connection transmitting signals representing a computer program according to the invention.

It should be noted that the above-mentioned embodiments illustrate rather than limit the invention, and that those skilled in the art will be able to design alternatives

20   without departing from the scope of the appended claims.

In the claims, any reference signs placed between parentheses shall not be construed as limiting the claim. The word 'comprising' does not exclude the presence of other elements or steps than those listed in a claim. The word 'a' is used as an equivalent of

25   'one or more' or 'at least one'.